# Système d'exploitation II

# TP 3 : Création Multiple et gestion des processus

## Partie II : Création Multiples et gestion des processus

Dans cette partie du TP nous traiterons la création multiple des processus, c'est-à-dire la mise en place d'une hiérarchie des processus sous linux. Avant d'étudier les procédures de gestion des permissions et d'envoi des signaux

**Question 1 :** Créer un programme C qui permet :

- La création de l'arborescence pour chaque figure
- L'affichage des attributs des processus

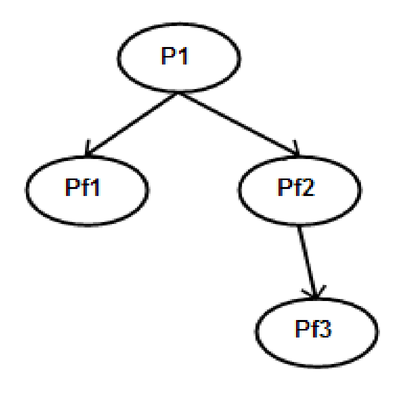



Figure 1 : Arborescence N°1

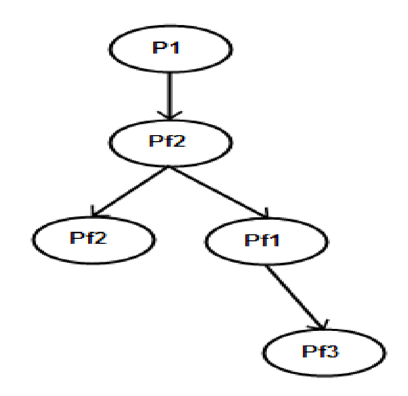



Figure2 : Arborescence N°2

**Question 2 :** Les priorités.

- Est-ce qu'on peut connaitre à l'avance l'ordre d'exécution ? Expliquer.

Sous linux on peut définir les priorités via la fonction **nice** et **renice**.

- **nice niveau commande** Pour définir la priorité d'un processus (une commande) avant son lancement.
- **renice niveau PID** Pour redéfinir la priorité d'un processus en cours d'exécution.

Avec, niveau est un entier qui définit la priorité, il prend des valeurs entre [-20, 20], un nombre plus petit signifie une priorité plus grande. Les valeurs négatives sont réservées à l'utilisateur root.

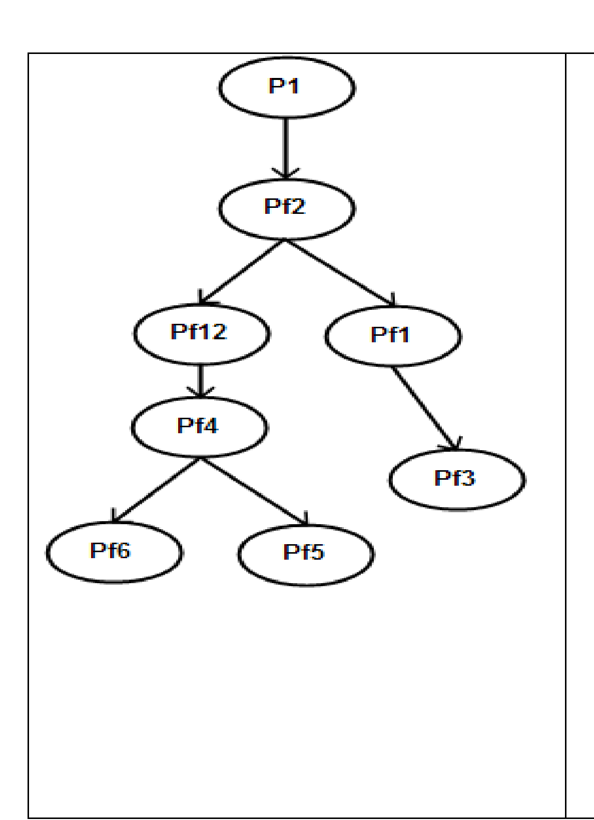
- Modifier le code pour que les processus attendent un instant avant de terminer l'exécution (utiliser la fonction sleep()).
- Rendre les priorités de PF3>PF2>PF1>P de la figure 1. Exécuter et expliquer.

**Question 3** : Arrêt d'un processus en exécution

En pratique, on est mené à tuer des processus en exécution pour une raison ou une autre. Pour cela, il suffit d'envoyer un signal au processus via la commande **kill,** qui permet l'envoi d'un signal au processus visé (utilisation du PID). La syntaxe est : # **kill -signal process-id**

Plusieurs types de signal existent sous linux parmi lesquelles :

- Définissez un signal. C'est quoi la différence entre les signaux et les pipes ? Donner un exemple de signal.
- Lister les signaux disponibles via la commande kill. Utiliser l'option –l.
- Créer un programme qui permet la création de l'arborescence 3, en affichant pour chaque processus son PID.

| | |
|---|---|
| <br> | • Modifier le code de la question c), pour rendre les processus pf12 et ces descendants infinis, c'est-à-dire ils s'exécutent via une boucle infinie.<br><br>• Exécuter le code, avant de lancer un signal d'interruption au processus pf12. Qu'est ce que vous remarquer ?<br><br>• Exécuter le code et envoyer un signal de type kill au processus pf2. Qu'est ce que vous remarquez ?<br><br>• Expliquer. |